6η Σειρά Ασκήσεων - Αλφαριθμητικά

Άσκηση 6.1

Να γραφεί μια συνάρτηση με όνομα `word_cnt` που να δέχεται ως είσοδο ένα αλφαριθμητικό μέγιστου μήκους `80` χαρακτήρων και να επιστρέφει το πλήθος των λέξεων που περιέχει. Οι λέξεις στο αλφαριθμητικό διαχωρίζονται μεταξύ τους με ένα κενό διάστημα.
Στη συνέχεια να γραφεί ένα πρόγραμμα που να χρησιμοποιεί τη συνάρτηση `word_cnt`. Η εισαγωγή των δεδομένων στη `main()` να γίνει με τη χρήση της `gets()`.
ΥΠΟΔΕΙΞΗ : μπορείτε να χρησιμοποιήσετε τη συνάρτηση `isspace` για να ελέγχετε αν ένας χαρακτήρας είναι το κενό διάστημα.

Άσκηση 6.2

Να γραφεί μια συνάρτηση με όνομα `reverse_string` που θα δέχεται ως είσοδο ένα αλφαριθμητικό μέγιστου μήκους `80` χαρακτήρων και θα επιστρέφει το αλφαριθμητικό ανεστραμμένο (π.χ. από το αλφαριθμητικό `“production & management”` θα προκύπτει `“tnemeganam & noitcudorp”`).
Στη συνέχεια να γραφεί ένα πρόγραμμα που να χρησιμοποιεί τη συνάρτηση `reverse_string`. Η εισαγωγή των δεδομένων στη `main()` να γίνει με τη χρήση της `gets()`.
Να διερευνήσετε τη δυνατότητα χρήσης δυναμικής παραχώρησης μνήμης.

Άσκηση 6.3

Χρησιμοποιώντας τις συναρτήσεις συμβολοσειρών που παρέχει η γλώσσα C για το χειρισμό αλφαριθμητικών (strings) να γράψετε ένα πρόγραμμα για το παρακάτω πρόβλημα:

1. Να εισάγετε από το πληκτρολόγιο ένα αλφαριθμητικό μέγιστου μήκους `80` χαρακτήρων με χρήση της `gets()`.
2. Να βρείτε και να εμφανίσετε τη συχνότητα εμφάνισης κάθε χαρακτήρα, με αλφαβητικό σειρά (δηλ. πρώτα τις συχνότητες των χαρακτήρων A,B,C,… και μετά τις συχνότητες των a, b, c,….). Για κάθε έναν από τους χαρακτήρες θα εμφανίζεται η τιμή του (δηλ. A, B, …) και δίπλα το πλήθος των εμφανίσεων του στο αλφαριθμητικό ως εξής :

```
A     2
B     0
.....
g     1
.....
```

ΥΠΟΔΕΙΞΗ : δεν πρέπει να χρησιμοποιήσετε την εντολή `if`. Να χρησιμοποιήσετε τις τιμές των

χαρακτήρων στον κώδικα ASCII δηλώνοντας κατάλληλα ένα μονοδιάστατο αριθμητικό πίνακα ακεραίων, με αρχικές μηδενικές τιμές, που θα περιέχει στο τέλος τις συχνότητες εμφάνισης.

Άσκηση 6.4

Να γραφεί μία συνάρτηση με όνομα `single_ch` που να έχει ως παράμετρο εισόδου ένα αλφαριθμητικό μέγιστου μήκους `80` χαρακτήρων. Η συνάρτηση να επιστρέφει ένα νέο αλφαριθμητικό που να περιέχει κατά σειρά όλους τους χαρακτήρες του αλφαριθμητικού εισόδου αλλά μόνον μία φορά τον καθένα δηλ. αν το αλφαριθμητικό εισόδου είναι `"engineering"` το νέο αλφαριθμητικό να είναι `"engir"`.
Στη συνέχεια να γραφεί ένα πρόγραμμα που να χρησιμοποιεί τη συνάρτηση `single_ch`. Η εισαγωγή των δεδομένων στη `main()` να γίνει με τη χρήση της `gets()`.

Άσκηση 6.5

Να γραφεί ένα πρόγραμμα το οποίο θα αντιγράφει ένα αλφαριθμητικό σε ένα νέο αλφαριθμητικό χωρίς να χρησιμοποιηθεί η συνάρτηση `strcpy()`.

Άσκηση 6.6

Να γραφεί ένα πρόγραμμα το οποίο θα βρίσκει εάν ένα αλφαριθμητικό είναι παλίνδρομο (δηλ. αν διαβάζεται το ίδιο από την αρχή προς το τέλος και από το τέλος προς την αρχή).

Άσκηση 6.7

Να γραφεί ένα πρόγραμμα το οποίο να προσομοιώνει το δημοφιλές παιχνίδι της κρεμάλας. Το πρόγραμμα να διαβάζει μια λέξη μέχρι 30 χαρακτήρες, η οποία θα αποτελεί τη μυστική λέξη. Στη συνέχεια, το πρόγραμμα να διαβάζει επαναληπτικά κάποιο γράμμα από τον παίκτη. Αν το γράμμα δεν περιέχεται στη μυστική λέξη, το πρόγραμμα να ενημερώνει τον παίκτη. Αν περιέχεται, να εμφανίζει τα γράμματα που έχουν βρεθεί και στα υπόλοιπα να βάζει τον χαρακτήρα της υπογράμμισης (“_”). Το παιχνίδι να ολοκληρώνεται όταν βρεθεί η λέξη ή αν ο παίκτης έχει κάνει 7 λανθασμένες προβλέψεις.
Σημείωση: Θεωρήστε ότι δεν επιτρέπεται ο χρήστης να εισάγει το ίδιο γράμμα πάνω από μια φορά.

Άσκηση 6.8

Ο Παγκόσμιος Κώδικα Προϊόντων (Universal Product Code - UPC) αποτελείται από 12 ψηφία. Το τελευταίο ψηφίο είναι το ψηφίο ελέγχου και χρησιμοποιείται για την ανίχνευση λανθασμένου γραμμοκώδικα (barcode). Για να υπολογίσουμε την τιμή του, χρησιμοποιούμε τα 11 πρώτα ψηφία ως εξής:

1. Προσθέτουμε τις τιμές των ψηφίων στις περιττές θέσεις και πολλαπλασιάζουμε το αποτέλεσμα με το 3.
2. Στο προηγούμενο αποτέλεσμα προσθέτουμε τις τιμές των ψηφίων στις άρτιες θέσεις.

3. Διαιρούμε το τελικό αποτέλεσμα με το `10`. Αφαιρούμε το υπόλοιπο από το `10` και αυτό είναι το ψηφίο ελέγχου. Αν το αποτέλεσμα της αφαίρεσης είναι `10` (που συμβαίνει όταν το υπόλοιπο είναι `0`), κάνουμε την τιμή του ψηφίου ελέγχου ίση με `0`.

Για παράδειγμα, ας ελέγξουμε αν ο γραμμοκώδικας: `123456789015` είναι έγκυρος.
Το ψηφίο ελέγχου υπολογίζεται ως εξής:

1. `1+3+5+7+9+1=26`. Πολλαπλασιασμός με το 3: `26*3=78`
2. `2+4+6+8+0=20`. Προσθέτουμε το προηγούμενο άθροισμα: `78+20=98`
3. Η τιμή του ψηφίου ελέγχου είναι: `10-(98%10)=10-8=2`

Αφού το τελευταίο ψηφίο είναι `5` και η τιμή του ελέγχου είναι `2`, ο γραμμοκώδικας δεν είναι έγκυρος.
Να γραφεί πρόγραμμα που να διαβάζει ένα γραμμοκώδικα και να ελέγχει αν είναι έγκυρος. Το πρόγραμμα να υποχρεώνει το χρήστη να εισάγει έναν έγκυρο γραμμοκώδικα, δηλαδή το μήκος του αλφαριθμητικού να είναι `12` και να περιέχει μόνο ψηφία.